2019-11-28

Identifying Good Algorithm Parameters in Evolutionary Multi- and Many-Objective Optimisation: A Visualisation Approach

Walker, David

10.1016/j.asoc.2019.105902
Applied Soft Computing
Elsevier

Identifying good algorithm parameters in evolutionary multi- and many-objective optimisation: a visualisation approach

David J. Walker and Matthew J. Craven

School of Engineering, Computing and Mathematics, University of Plymouth, UK.

Abstract

Evolutionary algorithms are often highly dependent on the correct setting of their parameters, and benchmarking different parametrisations allows a user to identify which parameters offer the best performance on their given problem. Visualisation offers a way of presenting the results of such benchmarking so that a non-expert user can understand how their algorithm is performing. By examining the characteristics of their algorithm, such as convergence and diversity, the user can learn how effective their chosen algorithm parametrisation is. This paper presents a technique intended to offer this insight, by presenting the relative performance of a set of EAs optimising the same multi-objective problem in a simple visualisation. The visualisation characterises the behaviour of the algorithm in terms of known performance indicators drawn from the literature, and is capable of visualising the optimisation of many-objective problems also. The method is demonstrated with benchmark test problems from the popular DTLZ and CEC 2009 problem suites, optimising them with different parametrisations of both NSGA-II and NSGA-III, and it is shown that known characteristics of optimisers solving these problems can be observed in the visualisations resulting.

Keywords: Benchmarking, Parametrisation, Visualisation, Multi-Objective, Many-Objective, Optimisation.

1. Introduction

Nature-inspired search algorithms are complex systems that are capable of optimising a great range of scientific and industrial problems. Though various approaches have been proposed, methods generally draw on the principles of natural selection and generate one or more solutions to a given problem. Such methods are comprised of *operators* that mimic a process observed in the natural world, which when assembled in a certain way result in a nature-inspired algorithm (such as an evolutionary algorithm (EA), particle swarm optimiser (PSO) or differential evolution (DE) algorithm). For example, the solution generation mechanism of a genetic algorithm (GA) usually comprises a crossover operation, combining genetic material from two or more solutions, followed by a mutation operation, in which a genetic change is introduced to a single solution. PSO requires the updating of position vectors of the members of a swarm to move the swarm

(representing a set of solutions) to a better part of the space, while DE requires the setting of a crossover rate and a step size. All of these example operators use parameters to control their operation, which the algorithm user must set in order to obtain the best possible results with their algorithm.

Parametrising an algorithm can be a significant challenge [6, 11], and can require expert knowledge about the use of nature-inspired techniques that is often not found in industry. This requirement of expert knowledge is likely to reduce the ease with which non-experts in industry can utilise cutting edge optimisation methods. This stems from the characterisation of many nature-inspired methods as *black-box* methods, meaning that the user cannot easily observe their operation in solving an optimisation problem. With this in mind, an obvious approach to making such techniques more readily understandable to people outside of the Evolutionary Computation (EC) field is to visualise the operation of the algorithms on a given problem in such a way that the user can identify operation characteristics that will assist them in the setting of parameters. Specifically, the aim of this paper is to visualise the benchmarking of different algorithm parametrisations so that the algorithm user can identify which parametrisations offer best performance for their problem.

As a subfield within EC, visualisation has been relatively active in recent years. Much of this activity stems from the fact that the visualisation of solution quality is often a non-trivial matter. Solution quality is quantified using an *objective function* – a mathematical formulation of solution quality. Problems comprising one such function are called *single-objective* problems, while those with more than one function are called *multi-objective*. A subset of these are termed *many-objective* problems, and comprise four or more objective functions. A considerable amount of research has been dedicated to visualising solutions to many-objective problems, as their objective values cannot be easily mapped into the two or three dimensions that a human decision maker can comprehend [30]. Some methods exist for visualising the search process itself [15, 19]; however, work in this area is much less advanced.

A method [27, 28] was recently proposed in which the performance of an EA operating under different parametrisations was visualised using performance indicators and colour to convey different aspects of algorithm performance. The method was demonstrated on simple continuous [27] and discrete [28] test problems, and it was shown that the method was able to convey known properties about the algorithm used on those problems. This paper presents an extension of these pilot studies, in which a more thorough characterisation of the visualisation method is made, and the information about performance that it can convey is enhanced. Specifically, the paper offers the following novel contributions:

1. The method is used to visualise the evolutionary behaviour of a greater range of algorithms. Both initial studies operated with a simple GA and illustrated the operation of its mutation operator; herein, the visualisation is expanded to show both crossover and mutation, and the problems are optimised with NSGA-II and NSGA-III.

2. Both initial studies considered multi-objective problems (comprising two and three objectives). This paper presents the example of many-objective problems, comprising five.
3. The initial studies used a limited set of performance measures (hypervolume and crowding distance). In this work, four indicators from the literature are examined – including indicators that characterise decision space as well as objective space.

The test problems used in this work are drawn from the well-known DTLZ problem suite [9] as well as the newer CEC 2009 suite [32]. The problems offered by the suite are scalable in the number of objectives, and provide a range of problem characteristics on which to test algorithms.

The remainder of the paper is structured as follows. Some relevant background material is discussed in Section 2, providing a brief introduction into multi- and many-objective optimisation and an overview of existing methods for visualising evolutionary processes. Section 3 introduces the visualisation method itself, before the experimental setup followed is outlined in Section 4 and results are discussed in Section 5. Concluding remarks are made in Section 6, as well as some pointers towards future avenues of research resulting from this work.

2. Background

The problems considered in this paper are continuous multi-objective problems, though it is noted that the visualisation method can generalise to discrete domain problems too [28]. Given a set of solutions to such a problem, the i-th solution is denoted by a D-dimensional vector of real numbers $\mathbf{x}_i$. This *decision space* component of a solution maps to an *objective space* component, an objective vector of M real values denoting solution quality. For a problem comprising M objectives, objective vectors are written as

$$\mathbf{y}_i = (f_1(\mathbf{x}_i), \ldots, f_M(\mathbf{x}_i)), \tag{1}$$

where y_{im} is the value of the i-th solution on the m-th objective. Relative quality between multi-objective individuals is compared using the *dominance* relation. Under dominance, solution $\mathbf{x}_i$ is superior to $\mathbf{x}_j$ if it is no worse than $\mathbf{x}_j$ on any objective and wholly better on at least one. Assuming a minimisation problem, $\mathbf{x}_i$ *dominates* $\mathbf{x}_j$ ($\mathbf{y}_i \prec \mathbf{y}_j$) if

$$\mathbf{y}_i \prec \mathbf{y}_j \iff \forall m\,(y_{im} \leq y_{jm}) \wedge \exists\, m\,(y_{im} < y_{jm}). \tag{2}$$

If neither $\mathbf{y}_i \prec \mathbf{y}_j$ nor $\mathbf{y}_j \prec \mathbf{y}_i$ then the two solutions are called *mutually non-dominating*. If $\mathbf{y}_i$ has no dominating solutions then the solution $\mathbf{x}_i$ is said to be *non-dominated*. As a multi-objective problem comprises a set of objectives which are usually in conflict (meaning that an optimal solution under one objective has a poor fitness under another objective), there is no single solution that simultaneously optimises all M objectives. Rather, the goal of optimising a multi-objective problem is to locate (or approximate) the *Pareto set*. This is the set of feasible solutions

that maps to a set of non-dominated solutions in objective space, called the *Pareto front*. Solutions lying on the Pareto front are mutually non-dominating.

Evolutionary methods have long been proposed that can generate a close approximation to the Pareto front, and cover the full extent of the front. More recently, a subclass of problems have been widely discussed in the EC literature – *many-objective* problems. These problems are characterised by four or more objectives, which cause several issues to affect standard multi-objective EAs (MOEAs). Of most relevance here is that the solutions generated by MOEAs have high-dimensional objective vectors, which cannot be visualised in the three spatial dimensions a human decision maker can comprehend using standard visualisation methods. Various approaches to resolving this have been discussed in recent years [30]. Another issue is that standard MOEA techniques cannot differentiate between high-dimensional objective vectors using dominance as the probability of generating a mutually non-dominating individual within the objective space, assuming a uniform distribution of objective vectors, increases rapidly with the number of objectives [12]. Algorithms such as MOEA/D [31] and NSGA-III [7] have been developed to optimise many-objective problems; NSGA-III is used later in this paper.

2.1. Benchmarking MOEAs and MaOEAs

Since the inception of multi-objective evolutionary algorithms, efforts have been made to benchmark them by reporting their performance on test problems against indicators. In a common workflow, a proposed new algorithm is run on several test problems, and its performance on those problems is assessed according to one or more indicators to describe, for example, the rate of convergence, coverage of the true Pareto front, and the extent to which diversity is preserved in decision space. Much of the early MOEA research was based on the use of test problems developed in isolation, before suites of test problems designed to expose algorithms to different problem features. The ZDT problems [33] provided researchers with a suite of 2-objective problems, before being surpassed by the DTLZ problem suite [9], which has become ubiquitous in MOEA research. Used herein, these problems are scalable in the number of objectives, and have underpinned the advent of MaOEAs (many objective EAs) in recent years. Other problem suites and frameworks have appeared, building on the strengths of their predecessors and providing problems that covered a greater range of problem features. Various reviews of problem suites have been published in the literature (e.g., [16]).

2.2. Visualising Evolution in Nature-inspired Methods

The identification of correct parameters for an EA operating on a specific problem instance has received considerable attention in the EC literature over the years. Important reviews have been published in [6, 10], and a myriad of approaches have been proposed to work with specific algorithms and on specific problems. A large majority of these methods are based on techniques such as *parameter sweeping* and the *a priori* identification of a tuning process such as the setting of an annealing schedule in a simulated annealing algorithm. In addition to selecting and

generating low-level heuristics, some hyper-heuristics also include parameter tuning elements, such as that proposed in [21]. While this paper is concerned with EAs, other parameter tuning in other types of metaheuristics has also been explored (for example, [26] discusses approaches to parameter tuning within ant colony optimisation). All of these approaches are automatic in nature – either an offline decision is made *a priori* that determines a parameter tuning policy that the algorithm follows, or an online parameter tuning approach is taken. These approaches differ to that taken in this work in that herein the aim is to include a decision maker – for example, the algorithm developer or a problem owner – such that they are tasked with identifying good algorithm parametrisations over poor ones using the visualisation proposed in this work. The purpose of this is to allow a decision maker to bring their own intuition about how the problem works so that they can better inform the optimisation process through interaction.

Of the work published on visualisation within EC, most focus falls on methods for visualising sets of solutions. As many-objective algorithms, capable of generating populations of solutions that are difficult to visualise, have matured it has been necessary to develop methods for presenting such populations to decision makers. Generally, population visualisation methods fall into three categories. The first includes techniques for presenting solutions in terms of the original objective vectors, such that objective values can be read from the visualisation. Examples include parallel coordinate plots [17] and heatmaps [23, 30]. In the second category, visualisations are constructed in terms of the full set of objectives, but the objective values are not observable from the visualisation [29]. Finally, objective vectors can be projected into a new low-dimensional coordinate space that can be presented with a standard visualisation tool such as a scatter plot [30]. Whatever category is employed, these visualisations are concerned with assisting the decision maker to navigate the trade-off between objectives, and therefore are not constructed in terms of the evolutionary processes used to generate solutions.

Various approaches have been taken to visualise algorithm operation. An example of them presented a visualisation of algorithm operation of an EA optimising the *water distribution network design problem* [19]. That problem requires the identification of a set of pipe diameters that provide optimality in terms of the overall cost of a water distribution network, and its hydraulic operation. Their approach coloured pipes according to the frequency with which they had been mutated, showing the user which regions of the network the algorithm had finished working on and which pipes were still being optimised.

Early visualisation work examined the visualisation of a single algorithm. An early example is [22], in which a set of tools for visualising algorithm state was proposed. Among the tools were visualisations for: displaying the history of a single run, showing all objective function values at all generations during the optimisation process; visualising the distance between decision vectors; and visualising the convergence properties of the algorithm. They also proposed using the Sammon Mapping [24] to execute a multidimensional scaling of the search space so that the relative locations of good solutions found during the search can be displayed. That work considered

only single-objective problems. GAVEL [15], also demonstrated on single-objective problems, provided another suite of tools with which evolutionary properties can be visualised. As well as providing the standard fitness visualisations showing convergence (or lack thereof) to the optimal point over time, GAVEL provides the ability to view the lineage of individual genes, as well as illustrating what type of evolutionary operators went into the construction of the solution. Another method proposed a tool called EAVis, which was applied to single-objective instances of the 0/1 Knapsack problem [20]. This enabled the user to view ranked solutions, in displaying both objective values for the population and individual decision vectors. Other information displayed included the origin of the fittest individual, including the ability to see all of the solutions from which it was derived.

Another sub-field within evolutionary computation in which lineage is important is that of genetic programming (GP). An example in which GP was visualised used a graph-based structure to visualise the origin of genetic material in the fittest final solution [3]. This enabled the algorithm user to identify which crossover operations (their algorithm did not perform mutation) generated useful solutions. Another study offered a similar capability, including mutation operations too [5].

One aspect relevant to this work is the study of algorithm parameter stability [4]. It is well known that various classes of optimisers are sensitive to changes in parametrisation, in that such changes may have either a productive or detrimental effect on algorithm success. In particular, a bad parametrisaton may result in no usable results at all. The work introduced a visualisation where, against a metric in parameter space mapped to a distance in the visualisation, algorithm performance is represented by colour. The visualisation acted as a tool to determine if an optimiser on a cryptology problem is stable or not with regards to perturbation size. The authors explored the question of how perturbing a parametrisation according to some metric changes algorithm performance, and provided some results about the boundary of stability in parameter space, i.e., the point at which algorithm performance "falls off a cliff".

3. The Proposed Visualisation

This paper demonstrates the use of a visualisation that displays algorithm parameters in combination with the performance of the algorithm, characterised in terms of the convergence and diversity of the search population. It does so in such a way that multiple algorithms are displayed alongside each other in order to aid their comparison, enabling them to be benchmarked by a non-expert user. The method is not concerned with solution representation; two pilot studies have been published, one demonstrating its use on continuous problems and a discrete industrial benchmark problem [27]; the other presented a lightly modified version of the original method and applied it to visualising the performance of MOEAs optimising water distribution system designs [28].

The original method operated in terms of individual solutions within a population, and placed them within a circle; the position of an individual within the visualisation is determined by its

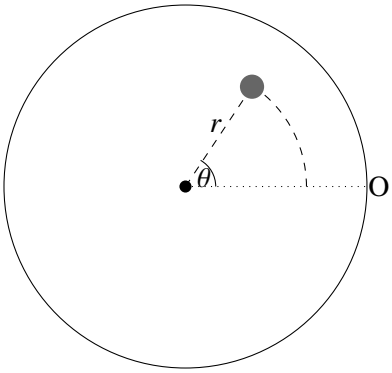


Figure 1: A schematic showing the placement of a single solution within the original visualisation method proposed. [27]

parametrisation, and its score against a chosen indicator of performance. The placement of a solution is shown in Figure 1. The single grey dot indicates a solution generated by one of the algorithms being benchmarked. The radius of the solution indicates the parametrisation; in the initial work, solutions were generated using a GA that operated with an additive Gaussian mutation which added a value drawn from a Gaussian distribution $\mathcal{N}(0, \sigma)$ to a randomly chosen decision variable. The value of σ was chosen uniformly in the range $[0, 1]$, and was the parameter shown using the radius of the solution. The parameter was fixed for an optimisation run, so a population of solutions was shown by a trail of dots at the same distance from the centre of the circle. The angle of the dot is determined by an indicator that conveys the quality of a solution. Standard approaches for evaluating the quality of an MOEA's operation are to consider its convergence and diversity properties, enabling the user to understand how close the population is to the true Pareto front, and whether there is sufficient diversity in the search population to allow it to properly search the feasible space, rather than collapsing in on a potentially sub-optimal region of the space.

The original studies measured convergence in terms of the hypervolume [13] (a rolling average was taken for the solution in question), while diversity was evaluated in terms of the crowding distance used to preserve diversity within NSGA-II [8]. That work considered small populations (10 solutions), which were found to produce good approximations to the Pareto front. When moving to consider the discrete benchmark problem, larger populations were required and the visualisation became cluttered. As such, a simple strategy was employed to "declutter" the visualisation. To this end, rather than displaying each solution, the extent of the arc occupied by a population was shown, along with the median indicator value. This was shown to substantially enhance the clarity of the visualisation.

The method proposed in this work is an extended version of the original method, and has been adjusted in two ways. First, in order to consider algorithms that have more than a single algorithm parameter the method has been extended to display multiple parameters. Secondly, an individual actor in the visualisation is now an entire population, rather than a solution. Figure

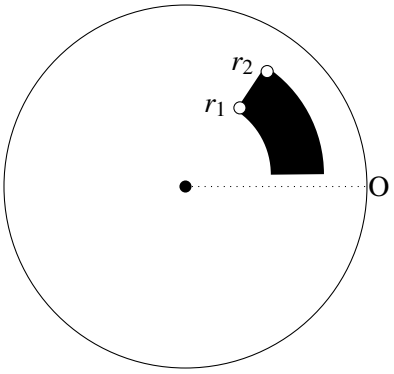


Figure 2: A schematic showing the placement of population within the extended visualisation proposed in this work.

2 demonstrates these extensions. As can be seen, whereas an individual was shown by a single dot, a population is shown by a shaded region. The region fills an arc of the circle, the extent of which illustrates the best and worst hypervolume score of any member of the population. The "front" of the population's region represents the best hypervolume, and the "back" of the region, which has moved the smallest distance from the origin line, represents the worst value. Two points are illustrated at the front of the region – these indicate the two parameter values (r_1 and r_2). Most of the visualisations herein do not explicitly highlight these, to prevent the visualisation becoming cluttered, however an interactive mode in which they can be activated is demonstrated. It is important to note that while this paper deals with an algorithm's requiring the setting of two parameters, any number of parameters could be shown in this way with the region defined in terms of the maximum and minimum parameter values used, with any other parameters being displayed on a continuum between them. The parameter value itself is read in the same way as was done in the original work, such that the distance from the centre of the visualisation shows the parameter value (all of the examples herein lie between 0 and 1, so an arc close to the centre has a parameter value of close to 0 and one near the perimeter of the circle has a value of close to 1).

3.1. Notation

In this work the following notation is used. In objective space, the aim of an optimisation algorithm is to create as small a distance, however measured, between an approximation set and a Pareto front. Though the true Pareto front is known for the test problems used in this work, it is demonstrated that the visualisation method can be used for real-world problems. To that end, assume that the true Pareto front is unknown and generate sample sets (which will be used as an "idealised" Pareto front). We note that in order to demonstrate the ability of the proposed visualisation to work with problems for which no Pareto front is known, we ignore the fact that the true front is known for the test problems used herein.

In order to generate sample sets, it is noted that R optimisation processes (in the results shown later in this paper, $R = 50$ simultaneous runs are used) are solving the same problem simultaneously, meaning that at any given generation g there are R populations of solutions. To generate a

sample set S_g, all R populations are combined and those individuals within this combined population that are non-dominated are identified. Those form the sample set for generation g. A set of solutions generated by an algorithm is denoted by R_g. An objective approximation set for generation g is denoted A_g (related to R_g via $A_g = f(R_g)$) and containing a given number of (approximations to) the objective vectors $\mathbf{y}_i$, each vector, of course, having M components.

Algorithm 1 Visualisation construction

```
1: for g ∈ (1, ..., G) do
2:   for r ∈ (1, ..., R) do
3:     for i ∈ (1, ..., N) do
4:       I_i^1 = indicator^1(y_i)
5:       I_i^2 = indicator^2(y_i)
6:     end for
7:   end for
8:   render_visualisation(max_{I^1}, min_{I^1}, Ī^2, p_1, p_2)
9: end for
```

The construction of a visualisation is summarised in Algorithm 1. For each g of the G generations a visualisation is produced that represents all R algorithm instances at generation g. Each of the N individuals in a population is assessed according to the two indicators, and the visualisation is rendered by providing the relevant information (the maximum and minimum values of the indicator controlling the position of the icon, the average of the second indicator that defines the icon's colour, and the two parameter values.)

3.2. Performance Indicators

One of the advantages alluded to in [27, 28] is the flexibility of the proposed method. As has been outlined above, the method requires the selection of two performance indicators – in the initial pilot studies this was restricted to the hypervolume and crowding distance. While it was shown that these measures provide a good indication of quality, other measures can be used in their stead. This paper considers four metrics.

Hypervolume: The hypervolume indicator [13] is a well-known indicator used to evaluate the comparative performance of MOEAs. It was used in the original demonstrations of the proposed visualisation method. Therein, the results were computed in terms of a reference point that was computed from the globally worst point found during any of the optimisers' execution. In this work, the initial populations of the algorithms are initialised so that the global worst point [14] can be used as the reference point and hypervolume scores can be normalised against it as the algorithms execute.

Crowding Distance: Crowding distance is a component of the NSGA-II algorithm, where it is used to break ties between members of the same non-dominated rank (i.e., solutions that are

incomparable under dominance) during selection. It operates by identifying the distance to its nearest neighbour in each objective, therefore providing a notion of diversity. While being used for selection, solutions with a large crowding distance are typically preferred as they reside in more diverse parts of the search space.

Objective Space Spread: This indicator was proposed in [1] to compute the spread of an approximation set (i.e., an approximation to the Pareto front) in objective space. Recall that the number of objectives in the optimisation problem is M. In this work the objective space spread is calculated for each generation g by the following:

$$I_s(g) = \left(\frac{\sum_{m=1}^{M} \left(\max_{\mathbf{y}_i \in A_g} (y_{im}) - \min_{\mathbf{y}_i \in A_g} (y_{im}) \right)^2}{\sum_{m=1}^{M} \left(\max_{\mathbf{z} \in S_g} (z_m) - \min_{\mathbf{z} \in S_g} (z_m) \right)^2} \right)^{1/2} \tag{3}$$

The scaling achieved by the denominator of (3) is performed with respect to the idealised Pareto front (Section 3.1). The resulting ratio $I_s > 0$ may be seen as simply the Euclidean length of the approximation set divided by that of the desired Pareto front in objective space. According to the authors, as I_s is effectively a multiplicative factor, a value of one represents the "best" diversity whereas a value of less than or greater than one represents "low" or "high" diversity, respectively.

Decision space diversity: Decision space diversity refers to the variety of solution in the search population, and is essential to ensuring a proper search of the problem search space. Various approaches have been taken to evaluate decision space diversity, though it tends to feature less in MOEAs and MaOEAs than measures for evaluating objective space diversity [18]. In terms of visualising diversity, a benchmark test problem was proposed that enabled the diversity of solutions to a many-objective problem to be considered [18]. In this work, a simple diversity measure is used wherein a scaled Euclidean distance between the individuals within the population is considered [25]. Diversity is computed as

$$D = \frac{2}{E \cdot N(N-1)} \sum_{i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|, \tag{4}$$

where E is the maximum Euclidean distance between any pair of solutions, and recall that $\mathbf{x}_i, \mathbf{x}_j$ are decision vectors and N is the population size. A high value indicates good diversity, and a low value indicates poor diversity. Note that herein decision vectors are formed of real numbers, which would not be the case for some other problem domains (such as discrete problems, for example). This method would easily adapt to such domains by replacing the Euclidean distance measure with an appropriate measure for the representation at hand.

4. Experimental Setup

The proposed visualisation method is demonstrated on benchmark multi- and many-objective optimisation problems from the literature, and visualises the optimisation of a GA. This section outlines the experimental setup followed.

In order to demonstrate the potential of the visualisation tool, it is demonstrated with data describing the optimisation of continuous test problems of differing complexity. Each test problem has its own characteristics that are designed to make the problem challenging (or not) in a specific way for an optimiser to solve. Test problems are used in the same way in this work: problems are selected that will show the optimiser working well in some cases, and poorly in others, so that the reader can observe the difference. The discussion around the example visualisations presented in Section 5 is structured in such a way that the visual properties of the visualisation are explained in the context of what they say about the optimisation process. The problems chosen are from the DTLZ problem suite [9], which were originally proposed to provide the EC community with test problems that were scalable in the number of objectives. It is important to note that the usage of this proposed visualisation method is not restricted to the DTLZ problem suite, but that it can be used with any multi-objective problem. The DTLZ problems were chosen for their extensive use in the literature. Herein, a 2-objective instance of the DTLZ2 test problem, 3-objective instances of DTLZ1 and DTLZ2, and a 5-objective instance of DTLZ2 are examined. Both DTLZ1 and DTLZ2 are continuous problems with decision variables lying in the region $[0, 1]$. The suggested number of decision variables is $D = k + M - 1$, where $k = 5$ for DTLZ1 and $k = 10$ for DTLZ2. The last k decision variables control the solution's distance from the Pareto front, and by construction the optimal value for these variables is 0.5. Thus, the true Pareto front is known.

In the same way as the proposed visualisation method can be used to illustrate the performance of an algorithm solving any problem, the method is also not restricted to use with a specific algorithm. The performance of any multi- or many-objective algorithm requiring the setting of parameters can be shown with this method. In this work, the efficacy of the proposed visualisation is demonstrated by optimising the problems described above with a GA. The initial study presented in [27] will be extended by considering the effect of different multiple perturbation operators – where the initial study focused on an additive Gaussian mutation operator, this paper illustrates the operation of a GA using both SBX crossover and Polynomial mutation.

As was the case in the initial pilot study, three variants of a GA are considered. While the same base algorithm is used for all three variants, with each using the same crossover and mutation strategy to generate solutions, each algorithm has a different selection operator. The algorithm begins by initialising a random population of solutions, which are evaluated under the problem objectives and used to initialise an archive of solutions. The archive will be used to store the current approximation of the Pareto front, and is unbounded in size. Each generation of the algorithm begins with the creation of a child population. This is done by creating child solutions with the SBX crossover operator that are then subjected to mutation with the Polynomial mutation

operator. In the case of both operators, the probability of perturbation (the crossover probability in SBX and the mutation probability in Polynomial mutation) are varied. The distribution index, controlling the size of the perturbation, in both cases is fixed (15 for SBX, 7 for polynomial mutation). Having generated a child, it is evaluated under the problem objectives and the archive is updated. If the child dominates any members of the archive then those members are deleted, and if the child is not dominated by any member of the archive then the new solution is added to it.

At the end of the generation, the parent population for the next generation is selected from the combined parent and child population of the current generation. As above, three selection operators are employed. The intention of this is to demonstrate different known characteristics for the chosen selection methods when applied to the benchmark problems used herein. The first uses non-dominated sorting, as is used in NSGA-II [8]. The population of objective vectors are sorted into non-dominated shells. In an iterative procedure, the non-dominated members of the population are identified, added to the next available non-dominated shell, and then are temporarily discarded from the population of objective vectors. This leaves a new set of non-dominated individuals, which become the next non-dominated shell, and so on until enough new parents have been selected. If the current non-dominated shell contains more solutions than are required to fill the population then the required number is sampled uniformly from that shell. This is included as a selection method that provides reasonable convergence to the Pareto front, as well as covering the front's full extent. The second method uniformly samples the combined parent and child populations, essentially reducing the selection pressure of the algorithm to that of a random walk through the space. This demonstrates an algorithm that does not converge in a reasonable time. The final method ranks the population with the average rank method [2]. Under average rank, the objective vectors of the combined parent and child populations are ranked by first converting the objective vectors to *rank coordinates*, by ranking the population M times, once by each objective. This places each objective on the scale $1, \ldots, M$, effectively normalising them, so that the rank of the i-th objective vector may be computed as

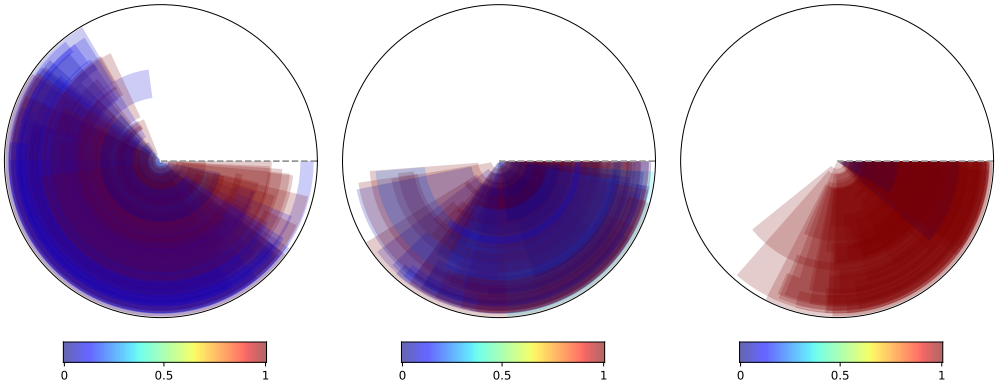
$$\hat{r}_i = \frac{1}{M} \sum_{m=1}^{M} r_{im}, \tag{5}$$

where r_{im} is the rank of the i-th solution on the m-th objective.

5. Results

5.1. Genetic Algorithm

Figure 3 presents a visualisation of the performance of 50 algorithm instances optimising a 2-objective instance of DTLZ2. These figures show three snapshots taken during the execution of the optimiser – the first at generation 1, the second at generation 10, and the final one at generation

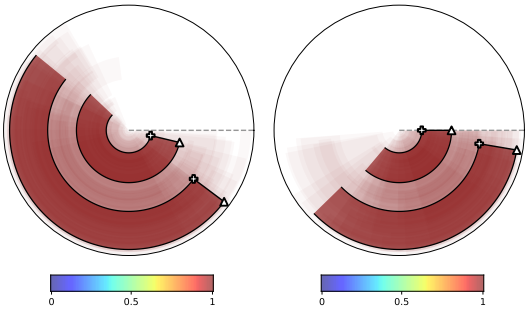


(a) Generation 1 (b) Generation 10 (c) Generation 49

Figure 3: Visualisations of the performance of NSGA-II optimising DTLZ2 in two objectives. The angle of the populations shown indicates convergence, as shown by the hypervolume while colour indicates population diversity, as shown by the crowding distance.

49. The optimiser was run for 5,000 function evaluations, which is known to be sufficient to optimise DTLZ2 in three objectives. Each block represents one of the 50 algorithm parametrisations used to optimise the problem and, as was explained in Section 3, the extent of the circle covered by an experiment's block indicates how well it has converged. As can be seen in generation 1, many of the parametrisations encompass approximately half of the extent of the circle. This indicates poor convergence, as would be expected at this phase of the optimisation process when the search population is comprised of random solutions. By generation 10, when 1,000 function evaluations have taken place, the populations have moved into the southern hemisphere of the circle, with many of them in the south-eastern quarter. This indicates that the optimisation process is driving the search population towards the true Pareto front as the hypervolume is increasing. In the final generation, shown in the right-hand plot, the search populations are almost exclusively in this south-eastern quarter. Those solutions with both low probabilities of crossover and mutation have converged particularly closely, and it can be seen that parametrisations wherein at least one of the algorithm parameters is large have performed less well. Colour indicates the crowding distance between solutions in the population – the median crowding distance is shown. As can be seen, there is a steady progression from a mixture of blue and red (indicating some solutions have a very small crowding distance and some have a larger crowding distance) indicating a preservation of diversity.

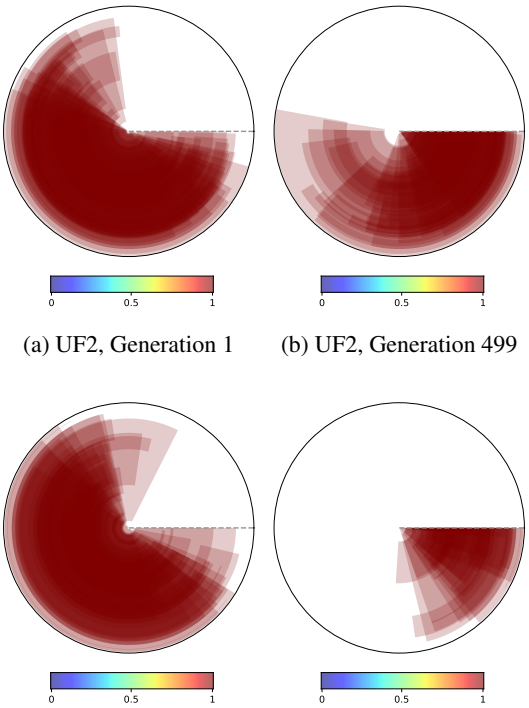
A potential issue with the visualisations shown in Figure 3 is that they could be cluttered and difficult to use for a non-expert user. To address this, Figure 4 illustrates a use case in which two of the algorithm instances have been highlighted. At the right-hand edge of the algorithm instances' region is a pair of icons, which represents the parametrisation of the algorithm instance. The plus icon indicates the crossover probability while the triangle indicates the mutation probability. As can be seen, the instance with the lower probabilities of crossover and mutation is doing much



(a) Generation 1 (b) Generation 10

Figure 4: Two snapshots showing the visualisation in an interactive mode, in which two algorithm instances have been highlighted.

better than the other instance. The solutions are better converged, and the diversity is better than that of the other instance. This is an example of the visualisation's use in an interactive setting – the user has identified two algorithm instances of interest and is now further exploring them using the tool. This facility could be easily expanded to include, for example, the selection of parameter ranges to view.



(a) UF2, Generation 1 (b) UF2, Generation 499

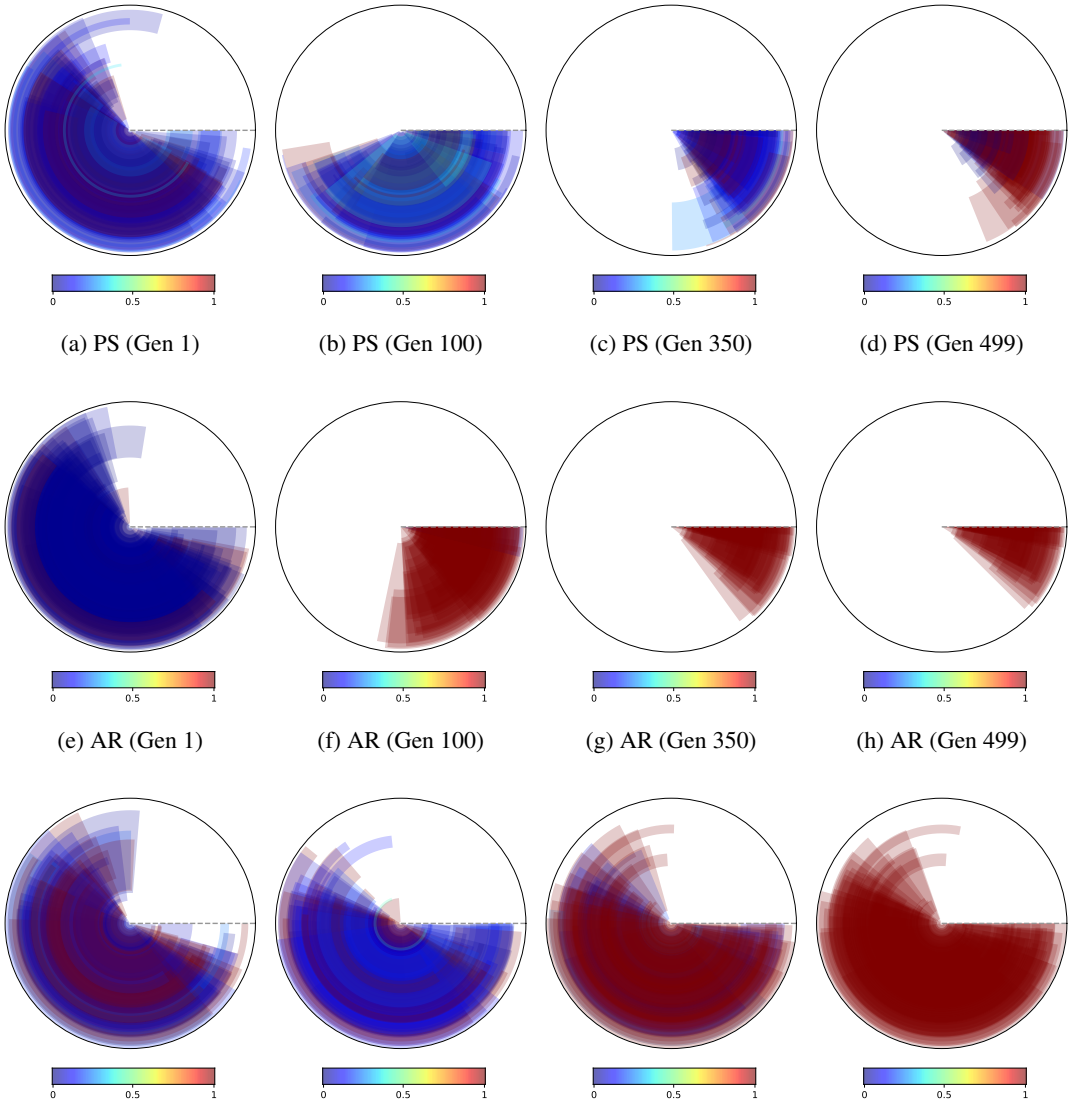
(c) UF6, Generation 1 (d) UF6, Generation 499

Figure 5: Visualisations of the optimisation of UF2 and UF6.

Results shown so far for the DTLZ2 test problem have illustrated a simple test problem. The method proposed is not dependent on a specific problem or suite of problems, and as Figure

5 illustrates its performance on problem drawn from a different problem suite. The CEC 2009 problem suite [32] was proposed to facilitate the benchmarking of algorithms on problems with more complicated search spaces than had previously been available to researchers. Two problems are shown used herein, both of which comprise 2-objectives. The top row illustrates the starting population and the final population of the UF2 problem, while the bottom row illustrates the corresponding generations of the optimiser solving the UF6 problem. UF2 comprises a search space in which the optimal solution set is difficult to locate and UF6's complexity is induced by a disconnected Pareto front. Both problems were optimised for 50,000 function evaluations. In the case of UF2, the algorithm has not converged. A substantial portion of the solutions are in the lower left-hand portion of the visualisation in generation 499, which indicates that the populations have not properly converged. The diversity measure (crowding distance) indicates that the algorithm instances are maintaining diversity. The results for UF6 show better convergence to the true Pareto front.

Figure 6 presents a set of results for the GA optimisation of a 3-objective instance of DTLZ1. The top row shows a set of optimisation runs for an algorithm using Pareto sorting selection; the middle row shows corresponding runs for an algorithm with average rank selection, and the bottom row shows an algorithm with a random selection operator. Each of the three algorithms has fifty algorithm instances (defined by a different parametrisation of the mutation operator). Four different points during the optimisation (generations 1, 100, 350 and 499) are shown. In the case of each algorithm, the generation 1 case is relatively similar; no optimisation has occurred at this point, so the populations are gathered above the origin line. By generation 100, shown in the second column, the Pareto sorting algorithm has begun to optimise the problem. This is shown by a set of populations that have moved into the southern hemisphere of the visualisation, showing that the hypervolume score increases for all of the algorithm instances. It can be seen from the visualisation that those algorithm instances with a smaller probability of crossover an mutation (i.e., those closer to the centre of the visualisation) are progressing faster around the extent of the visualisation than those at the outside – in all cases the worst performing algorithm instance is again one with high values for both parameters, seen by the lagging block at the periphery of the circle. This shows that, in the case of this problem, there is a slight advantage to optimising with smaller perturbations and exploiting good solutions rather than making large perturbations away from them. By generation 350, the search has progressed into the south-eastern quarter of the visualisation, indicating good convergence. Note that the search appears to have stagnated between generations 350 and 499, with little progress between the two. The change in colour between the two generations indicates that the population has stopped searching (it has converged as far as it will towards the Pareto front) and has then exploited this position to cover the full extent of the front, causing the crowding distance to become more uniform and eliminating extremely close solutions. At first glance, it appears that the population has not converged to the Pareto front; this is not so. In fact, this is a symptom of the premature convergence that is known to occur



(a) PS (Gen 1) (b) PS (Gen 100) (c) PS (Gen 350) (d) PS (Gen 499)

(e) AR (Gen 1) (f) AR (Gen 100) (g) AR (Gen 350) (h) AR (Gen 499)

(i) RS (Gen 1) (j) RS (Gen 100) (k) RS (Gen 350) (l) RS (Gen 499)

Figure 6: Optimisation traces for three optimisers solving a 3-objective instance of DTLZ1. Convergence is shown by the populations' hypervolume while diversity is shown by the crowding distance.

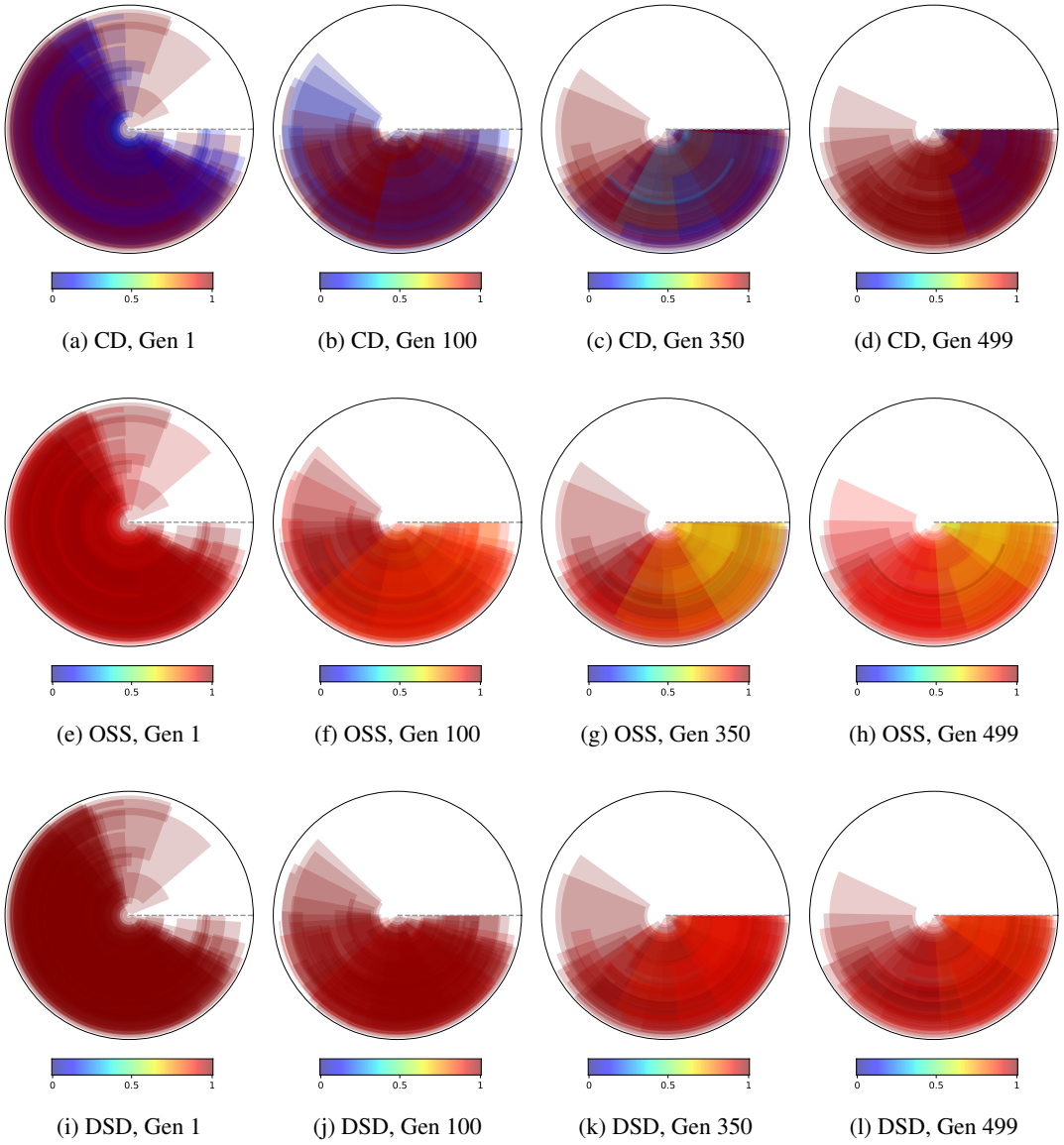
when using average rank as a selection operator on its own. The average rank operator prefers solutions that are highly ranked on multiple objectives – these solutions occur in the corners of the population, and the search therefore concentrates in those areas, leaving the majority of the true Pareto front unexplored [14]. This problem is exacerbated by the DTLZ problems' bias towards the top of the Pareto front. Therefore, in this case, the rapid convergence toward the Pareto front is shown by the populations' quick movement through the extent of the circle, and the population's diversity collapse is indicated by the early change in colour from blue to red, which persists for the remainder of the optimisation. This problem might be partially addressed by constructing visualisations in terms of the archive rather than the search population, however it is important to

note that the archive is often unbounded, and as such can potentially contain a substantially larger number of solutions than are found in the bounded population. While this is not necessarily a problem in this work, recall that the eventual aim is to include this method within an optimisation process to visualise the process online. In such a case, it would be computationally inefficient to render the large number of solutions in an archive. Additionally, while the archive maintains a wider range of solutions, these are the solutions that have not been dominated by any others generated by the optimisation process, and since the bias induced by the problem and selection operator quickly move the search away from these additional points they are likely to be somewhat suboptimal.

From examining the central row, showing average rank optimisation, it can be seen that the populations have converged rapidly, with all populations residing in the south-eastern quadrant by generation 100. Interestingly, the crowding distance indicates that the solutions have spread as far as they will during the search as the progression from blue to red is much more rapid. This indicates premature convergence, which is a known artefact of utlising an average rank-based selection operator wherein the operator ranks corner solutions highly and a bias is therefore induced that ignores a substantial part of the rest of the Pareto front [14]. Considering the central row, the random selection operator (unsurprisingly) does not optimise the problem at all. At each generation, a set of child solutions are generated from the current parent population, and then the parent population for the next generation is selected by choosing N (in this case, $N = 100$) from the union of the current parent and child populations. As such, there is no selection pressure exerted by the selection operator, and the search process never really begins.

5.2. Visualising diversity

Throughout the examples shown so far diversity has been demonstrated with the crowding distance measure. Figure 7 illustrates four snapshots during the execution of NSGA-II (using Pareto sorting for selection) for a 3-objective instance of DTLZ2. Diversity is shown in three ways. The top row illustrates crowding distance, as has been shown previously. The second row shows the objective space spread indicator described earlier, while the bottom row shows the decision space diversity measure. As before, population angles are determined according to hypervolume. The crowding distance results follow a similar pattern to those shown earlier – the diversity within the population normalises as the optimisation procedure progresses. An interesting artefact of the objective space spread measure is that as the objective values reduce (as the solutions minimise) so the diversity measure is shown to reduce. This can be seen as the colour progresses from red to yellow in later generations. A similar effect can be seen when considering decision space diversity, though to a lesser extent. This reduced change over time indicates that the decision space diversity does not alter significantly during the optimisation process, which is intuitively correct as the search space for DTLZ2 is known to be relatively simple. Revealing different characteristics with different metrics highlights the importance of



(a) CD, Gen 1 (b) CD, Gen 100 (c) CD, Gen 350 (d) CD, Gen 499

(e) OSS, Gen 1 (f) OSS, Gen 100 (g) OSS, Gen 350 (h) OSS, Gen 499

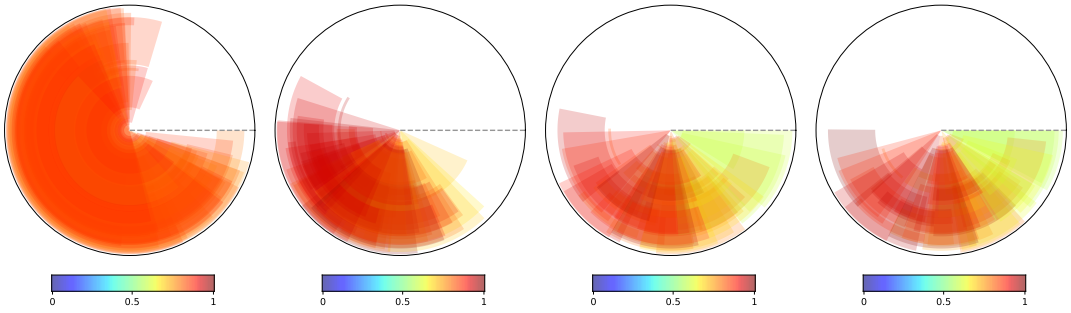
(i) DSD, Gen 1 (j) DSD, Gen 100 (k) DSD, Gen 350 (l) DSD, Gen 499

Figure 7: Visualisations of the 3-objective DTLZ2 optimisations, this time coloured according to different measures of diversity. The top row (a)–(d) are coloured according to crowding distance. The middle row (e)–(h) are coloured according to the objective space spread measure, while the bottom row (i)–(l) are coloured according to the decision space diversity metric.

having a suite of metrics available for the user to explore their data with. The effect of switching between the metric used to colour solutions is computationally inexpensive, and so can be done interactively as the optimisation progresses.

5.3. Many-objective Problems

The examples presented so far have been multi-objective – problems comprising two or three objectives. As many-objective problems have been explored, so algorithms capable of optimising them have been proposed. The final examples shown herein demonstrate the use of the proposed



(a) Gen 1 (b) Gen 20 (c) Gen 45 (d) Gen 65

Figure 8: Visualisations of the performance of NSGA-III optimising a 5-objective instance of DTLZ2.

method for visualising the performance of NSGA-III for optimising a 5-objective instance of DTLZ2. Part of the motivation behind the design of DTLZ2 is to demonstrate an algorithm's ability to scale, so it is well suited to use in this way.

Figure 8 illustrates the result of visualising the performance of using NSGA-III to generate a set of solutions to the problem. The results are of interest in two ways. First, despite being run for longer (up to 65 generations are shown) the optimiser has failed to converge. At the end of the procedure the majority of search populations are still encompassing the entire southern hemisphere of the circle. Secondly, the diversity is shown to be dropping much more markedly than has been seen in the results shown elsewhere in the paper. This indicates that the optimiser is preferring specific areas of the Pareto front; NSGA-III is a decomposition-based optimiser, so this makes intuitive sense, and the visualisation therefore provides a view on how the optimiser is functioning.

6. Conclusions

Visualisation of algorithm operation is an important aspect of research within the EC community, as it will help to lower the amount of information required by an algorithm user to employ EAs within real-world scenarios. This paper has presented an example of such a visualisation, allowing a user to observe different parametrisations of their algorithm. The method extends a visualisation tool proposed recently by expanding the number of algorithm parameters that can be visualised with it. Additionally, a greater range of performance indicators have been considered, the lightweight nature of which is better suited to the online visualisation that is the eventual goal of this work. The extension of the work to include many-objective optimisation further extends the applicability of the proposed method.

A principle advantage of the proposed method is its flexibility; the method will operate with solutions of any representation. As has been demonstrated, the method can work with any performance indicator that results in a single value evaluating the quality of a population. Due to that, the user can replace the indicators shown herein with any indicator that better suits their purpose.

This is particularly beneficial given the different representations that the method can be used with, as the user might wish to select, for example, a diversity measure that describes the behaviour of the algorithm in decision space. Other types of indicator that might be used include generational distance and inverted generational distance. We note that both would require the presence of a reference set, which may preclude their use with real-world problems for which no reference set is available. It may be possible to use a strategy such as that employed herein with the objective space spread measure proposed by Adra [1]. An additional element of flexibility is that, in its redesigned form, the method does not require a fixed number of algorithm parameters. Examples shown herein have been for algorithms comprising two parameters, though there is no computational reason that more could not be included. That said, the resulting visualisations are likely to become cluttered and difficult to read. Despite the potential to address this with interaction, allowing the user to disregard extraneous information and enhancing the clarity of the visualisation in the areas they are particularly concerned with, it is for that reason the method is currently likely to be more beneficial for algorithms with small numbers of parameters.

The method in its current form provides a useful way in which the value of different algorithm parametrisations can be observed. In its offline state, the method offers some utility, however its worth will be greatly expanded by facilitating its inclusion in an online tool. This will offer algorithm users the ability to observe which parametrisations are not generating strong solutions as the algorithm executes, and those instances that are not useful can be stopped – releasing valuable computing resources that can potentially be better spent on instances that are showing promise. An important aspect of this work will be to consider the performance of an interactive optimiser that utilises this approach compared to the automatic parameter tuning approaches discussed in Section 2. Work is ongoing to this end; at the moment, the visualisation tool is reasonably lightweight, however used in its current form the tool would use up the resources that it is designed to save. As such, an avenue of current investigation is the method's redevelopment to use GPUs, alleviating the additional work that would otherwise be done by the machines performing the optimisation itself. This is particularly important in the case of many-objective problems, where the number of solutions required to cover the extent of a high-dimensional Pareto front means that the performance indicator evaluation is slower. Additional further work entails exploring potential approaches to incorporating more information into the visualisation, both in terms of the algorithm parameters that can be shown and the number of performance indicators that can be used to illustrate performance.

7. Acknowledgements

The authors would like to thank Dr Alma Rahat for useful discussions on aspects of this work.

References

References

[1] S. F. Adra and P. J. Fleming. Diversity management in evolutionary many-objective optimization. IEEE Transactions on Evolutionary Computation, 15(2):183–195, April 2011.

[2] P. J. Bentley and J. P. Wakefield. Finding acceptable solutions in the Pareto-optimal range using multiobjective genetic algorithms. Soft Computing in Engineering Design and Manufacturing, pages 231–240. Springer, 1998.

[3] B. Burlacu, M. Affenzeller, M. Kommenda, S. Winkler, and G. Kronberger. Visualization of genetic lineages and inheritance information in genetic programming. Proceedings of Visualisation in Genetic and Evolutionary Computation (VizGEC 2013), held at GECCO 2013, pages 1351–1358. ACM, 2013.

[4] M. J. Craven and H. C. Jimbo. EA stability visualization: Perturbations, metrics and performance. Proceedings of Visualisation in Genetic and Evolutionary Computation (VizGEC 2014), held at GECCO 2014, pages 1083–1090. ACM, 2014.

[5] A. Cruz, P. Machado, F. Assunção, and A. Leitão. Elicit: Evolutionary computation visualization. Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation, pages 949–956. ACM, 2015.

[6] Kenneth De Jong. Parameter Setting in EAs: a 30 Year Perspective, pages 1–18. 2007.

[7] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based non-dominated sorting approach, part i: Solving problems with box constraints. IEEE Transactions on Evolutionary Computation, 18(4):577–601, Aug 2014.

[8] K Deb, A Pratap, S Agarwal, and T Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Transactions on Evolutionary Computation, 6(2):182–197, Apr 2002.

[9] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. volume 1 of Proceedings of IEEE Congress on Evolutionary Computation, pages 825–830. ACM, May 2002.

[10] A Eiben, Zbigniew Michalewicz, and Marc Schoenauer. Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 3:19–46, 2007.

[11] A. E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 3(2):124–141, July 1999.

[12] M. Farina and P. Amato. On the optimal solution definition for many-criteria optimization problems. Proceedings of the 2002 Annual Meeting of the North American Fuzzy Information Processing Society, pages 233 – 238. IEEE, 2002.

[13] M Fleischer. The Measure of Pareto Optima. Applications to Multi-objective Metaheuristics. EMO 2003 - 2nd International Conference in Evolutionary Multi-Criterion Optimization, pages 519–533. Springer, 2003.

[14] M. Garza-Fabre, G. Toscano-Pulido, and C. A. Coello Coello. Two novel approaches for many-objective optimization. Proceedings of IEEE Congress on Evolutionary Computation, pages 4480–4487. ACM, July 2010.

[15] E. Hart and P. Ross. GAVEL - a new tool for genetic algorithm visualization. IEEE Transactions on Evolutionary Computation, 5(4):335–348, Aug 2001.

[16] Simon Huband, Phil Hingston, Luigi Barone, and Lyndon While. A review of multi-objective test problems and a scalable test problem toolkit. IEEE Transactions on Evolutionary Computation, 10(5):477–506, October 2006.

[17] A. Inselberg. Parallel Coordinates: Visual Multidimensional Geometry and its Applications. Springer, 2009.

[18] H. Ishibuchi, N. Akedo, and Y. Nojima. A many-objective test problem for visually examining diversity maintenance behavior in a decision space. Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, pages 649–656. ACM, 2011.

[19] E. Keedwell, M. Johns, and D. Savić. Spatial and temporal visualisation of evolutionary algorithm decisions in water distribution network optimisation. In Proceedings of Visualisation in Genetic and Evolutionary Computation (VizGEC 2015) held at GECCO 2015, GECCO Companion '15, pages 941–948, 2015.

[20] A. Kerren and T. Egger. EAVis: A Visualisation Tool for Evolutionary Algorithms. Proceedings of 2005 IEEE Symposium on Visual Languages and Human-Centric Computing, pages 299–301. IEEE, 2005.

[21] A. Kheiri and E. Keedwell. A sequence-based selection hyper-heuristic utilising a hidden markov model. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2015), 2015.

[22] H. Pohlheim. Visualization of evolutionary algorithms – set of standard techniques and multidimensional visualization. Proceedings of Genetic and Evolutionary Computation Conference (GECCO 1999), pages 533–540. ACM, 1999.

[23] A. Pryke, S. Mostaghim, and A. Nazemi. Heatmap visualization of population based multi objective algorithms. Evolutionary Multi-Criterion Optimization (EMO 2006), pages 361–375. Springer, 2006.

[24] J. W. Sammon. A nonlinear mapping for data structure analysis. IEEE Trans. Comput., 18(5):401–409, 1969.

[25] O. M. Shir, M. Preuss, B. Naujoks, and M. Emmerich. Enhancing decision space diversity in evolutionary multiobjective algorithms. Proc. EMO 2009, pages 95–109. Springer, 2009.

[26] Thomas Stützle, Manuel López-Ibán ez, Paola Pellegrini, Michael Maur, Marco Montes de Oca, Mauro Birattari, and Marco Dorigo. Parameter adaptation in ant colony optimization. Technical Report TR/IRIDIA/2010-002, IRIDIA, January 2010.

[27] D. J. Walker and M. J. Craven. Toward the online visualisation of algorithm performance for parameter selection. International Conference on the Applications of Evolutionary Computation, pages 547–560. Springer, 2018.

[28] D. J. Walker and M. J. Craven. Visualising the operation of evolutionary algorithms optimising water distribution network design problems. Proceedings of the 13th International Conference on Hydroinformatics, pages 2250–2258. 2018.

[29] D. J. Walker, R. M. Everson, and J. E. Fieldsend. Ordering and visualising many-objective populations. 2010 Congress on Evolutionary Computation, pages 3664–3671. IEEE, 2010.

[30] D. J. Walker, R. M. Everson, and J. E. Fieldsend. Visualising mutually non-dominating solution sets in many-objective optimization. IEEE Transactions on Evolutionary Computation, 17(2), April 2013.

[31] Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation, 11(6):712–731, Dec 2007.

[32] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, Wudong Liu, and Santosh Tiwari. Multiobjective optimization test instances for the cec 2009 special session and competition. Technical Report CES-487, University of Essex, UK, Nanyang Technological University, Singapore and Clemson University, USA, 2008.

[33] E Zitzler, K Deb, and L Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation, 8(2):173–195, 2000.